IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application of

Allan Scherr

for

Network Accelerator

Cross Reference to Related Applications

This application is a continuation of copending U.S. Patent Application Serial No. 09/705,114 filed November 2, 2000 which is a continuation of U.S. Patent Application Serial No. 09/659,482 filed April 20, 1998 (now abandoned), which applications are assigned to the same assignee as this invention.

Background Of The Invention

Field of the Invention

The invention relates generally to the field of networking and in particular to the field of using auxiliary storage systems such as disk drives as caches for performance improvements in networks.

Background

As more users and more websites are added to the World Wide Web on the Internet, the content of the information transmitted on it also increases in complexity and quantity: Motion video, more complex graphics, audio transmissions, and so on, place rapidly increasing performance demands on the Internet at all points. The problem faced by service and content providers as well as users is how to maintain or improve performance for a growing user base without constantly

creating the need for additional capacity or "bandwidth" in the network.

Websites and web browser software such as provided by Netscape Communications Corporation (having a principal place of business in, Mountain View, California on the World Wide Web (WWW), use storage systems such as magnetic disks to store data being sent and received, and most of these also use a simple form of disk caching at the website or at the user site to improve performance and minimize re--transmissions of the same data. These typically use a "least recently used" (LRU) algorithm to maintain the most recently referred to data in the disk cache and a protocol that permits a user to request that a page be refreshed even if it is in the cache. However, as the traffic continues to grow, this method needs to be improved upon to provide the performance that may be required.

Traffic increases as subsequent requests are made for web pages that had been sent earlier, but are no longer in the local user's system. The same re-transmission will occur at other points in the network, thus degrading overall response time and requiring additional network bandwidth. One approach that is frequently used to tackle the problem is the use of faster transmission media to increase bandwidth. This takes large capital and labor expense to install and may also require

2

replacement of modems and other equipment at various nodes. Service providers that install faster transmission equipment must still match the speeds at which their users can send and receive data, thus bottlenecks can still occur that slow down performance and response times at the user's site.

Users who upgrade to faster transmission media may often have to scrap modems and other units that were limited to slower speeds. Somewhat less frequently, large-scale internal network wiring changes may need to be made, as well, often causing disruptions to service when problems are found during and after installation. With any of these changes, software changes may also be required at the user's site, to support the new hardware.

Despite the users' best efforts, a well-known phenomenon in network systems design, called the "turnpike" effect, may continually occur as users upgrade to faster transmission media. As United States interstate highway builders first observed in the 1950's, when better, "faster" highways were made available, more people tended to use them than were initially anticipated. A highway might have been designed to handle a specific amount of traffic, based on then present patterns and data. But once people learned how much faster and smoother travel on the new highway was, traffic might increase

to two or three times the original projections, making the highway nearly obsolete almost at the outset of its planned life.

Similar problems occur with users of the Internet and service and content providers. Many of the service providers and online system services have had difficulty adding systems and transmission links to keep up with such increases in traffic. As technology improves in all areas, content providers are providing more graphics, videos and interactive features that impose major new loads on the existing transmission systems. As companies and institutions install or expand local and wide area networks for their internal use, they are also linking them to Internet providers and sites, usually through gateways with "firewalls" to prevent unauthorized access to their internal networks. As these companies link their internal networks to the Internet and other external networks, usage and traffic on the Internet increases multi-fold. Many of these same companies and institutions are also content providers, offering websites of their own to others.

The content providers add to the problem of increased traffic in yet another way, when time-sensitive data is stored and transmitted. Stock quotes, for example, during the hours when a given exchange is open, are highly time sensitive. Web

pages containing them or other market information need to be
updated frequently during trading hours. Users who are tracking
such quotes, often want to insure that they have the latest
update of the web page. If standard Least Recently Used (LRU)
5   caching algorithms are used at the user site and this web page
is in constant use, the cached copies may not be refreshed for
several cycles of stock price changes: Here, caching data works
to the user's disadvantage.

However, once that exchange closes, there should be no
10   updates until the following business day. For the high-volume,
high-visibility exchanges, this means traffic can reach peaks
of congestion during trading hours. The network capacity used
to keep up with this may lie dormant during off-peak hours.
Most existing service and content providers on the Internet do
15   not, at present, have an effective way to differentiate between
these service levels in their prices or service offerings.

Private dial-up services, such as Westlaw™ from West
Group, having a principal place of business in Eagan,
Minnesota, or Lexis/Nexis™ of the Reed Elsevier PLC group
20   having a principal place of business in Dayton, Ohio or
Compuserve™ or America Online™ of America Online,
Incorporated, having a principal place of business in Dulles,
Virginia have been able to offer differentiated pricing for

networked access to certain kinds of data in their proprietary

databases, but doing this is greatly simplified when the

choices are limited and relatively few in number. In most cases

this is done on the basis of connect time and perhaps some

5   additional fee per database accessed.

Data management methods, such as least recently used

caching, can be applied to proprietary databases as well.

Usually only one form of data or cache management is associated

with a database, and the choice of a particular method of data

10   and cache management has historically been based on the type of

file being created.

On the Internet, by contrast, data requests can come from

anywhere in the world for almost any topic in the world, to any

content provider in the world. Patterns of access and

15   timeliness requirements vary greatly from user to user. An

educational institution that provides Internet services to its

students and faculty will have one set of needs for access, and

response times, while a business corporation user may have a

completely different set of needs.

20   Access to data on the Internet also differs from dial-up

access to proprietary databases in another way. The private

dial-up service provider may not change the services offered

for months or even years at a time. Data files may be updated,

but the kinds of information that can be obtained may remain constant.

On the Internet, the opposite is true. Information that was not available three months ago anywhere in the world may now be available from several different sources. This is also true for the format of the information. In less than a three year time span, web pages have gone from text only, to text plus drawings, then to text plus high-resolution photographic-like images in several different formats. Sound is also available now from many sites. Web browsers now permit use of videos and interactive forms. Traditional network and data management techniques are hardpressed to keep up with these changes.

It is an object of the present invention to provide a method and apparatus for improving network response time at one or more sites or nodes while reducing the amount of bandwidth used to carry a given load.

Another object of the present invention is providing improvements in network response time without requiring any changes in transmission media and transmission equipment.

Still another object of the present invention is providing a flexible method and apparatus for providing response time

7

improvements that can readily be adjusted to different usage patterns.

A further object of the present invention is providing a method and apparatus that permits a service or content provider to offer differentiated levels of service and prices based on the type of data being transmitted.

Summary of the Invention

These and other objects are achieved by a network accelerator storage caching system that may be inserted at any point in a network, to provide a configurable, scalable variety of cache management systems to improve response time. Depending on the configuration(s) selected, the system may manage data or subsets of data in a storage cache on the basis of time-currency, page usage frequency, charging considerations, pre-fetching algorithms, data-usage patterns, store-through methods for updated pages, least recently used method, B-tree algorithms, or indexing techniques including named element ordering, among others. A preferred embodiment may embed the configurable cache management in the storage media, either as firmware in a storage controller or as software executing in a central processing unit (CPU) in a storage controller. In a preferred embodiment the system may be scaled in size and offer security for protected data.

It is an aspect of the present invention to provide improvements in response times.

It is another aspect of the present invention to reduce the bandwidth required in the vicinity of the invention to transmit information responsively.

Another aspect of the present invention is to enable configuring at each site to use the cache method(s) preferred by that site.

A further aspect of the present invention is allowing a site to trade storage space for transmission capacity or bandwidth.

Brief Description of the Drawings

Figure 1a is a schematic drawing of various sites on a network using the present invention.

Figure 1b is a schematic drawing of illustrative embodiments of the present invention.

Figure 2a is a flow diagram of the present invention.

Figure 2b is a flow diagram of the configurator of the present invention.

Figure 3 is a flow diagram of a least recently used cache management method used in the present invention.

Figure 4 is a flow diagram of a time-sensitive method of cache management used in the present invention.

9

Figure 5 is a flow diagram of a data usage cache management method used in the present invention.

Figure 6 is a flow diagram of a pre-fetch cache management method used in the present invention.

Figure 7 is a flow diagram of a charging cache management method used in the present invention.

Figure 8 is a flow diagram of a B-tree cache management method used in the present invention.

Figure 9 is a flow diagram of an indexed cache management method used in the present invention.

Figure 10a is a flow diagram of a store-through method of cache management used in the present invention. Figure 10b is a flow diagram of a data protection method according to the present invention.

Figure 11 is a block diagram of scripted variables and pseudo-code for a pre-fetch method of cache management used in the present invention.

Figure 12 is a block diagram of scripted variables and pseudo-code for a time sensitive method of cache management used in the present invention.

Figure 13 is a table showing the elements of a Uniform Resource Locator (URL).

10

Figure 14 is table showing some of the named elements that can be included in HTML pages.

Figure 15 is a schematic drawing of a form used in the present invention.

Detailed Description of the Invention

In Figure 1a, a number of network sites using the present invention is shown. In a preferred embodiment, cache management system 10, includes a control device 12, and storage units 14. Control device 12, in this preferred embodiment, includes firmware that executes the logic of the present invention. Cache management system 10 is shown here installed at various sites on an Internet network. For purposes of illustration, a service provider site 00 is shown connected by transmission media T1 to a backbone link site 04. One or more backbone link sites 04 may be used for sending and receiving messages through the network. Local site 06 is shown here connected to the network formed by one or more backbone links 04 via transmission media T2. Local site 06 might be a corporate firewall & gateway site connected to user stations 08 inside an internal corporate network with a local area network as transmission media T3 or it could be a local service provider providing dial-up services to user stations 08 over

transmission media T3. Also shown in this Figure 1a, is a content provider site 02.

In a preferred embodiment, as shown in Figure 1b, storage unit 14 of cache management system 1ob is a large magnetic recording disk array, such as a redundant array of independent disks (RAID) system or multiple RAID systems installed at the site. A preferred embodiment might use even larger disk arrays such as EMC Corporation's (of Hopkinton, Mass.) Symmetrix™ having as much as 1.1 gigabytes of storage for large backbone link sites 04.

As will be apparent to those skilled in the art, other types of fast random access storage media can be used as storage units 14, such as magneto-optical disks, or massive random access memory arrays.

In a preferred embodiment, cache management system 10 can be scaled up or down in storage capacity to meet site requirements. Similarly, in a preferred embodiment of the present invention;-control device 12 is the controller for the disk system, where such controller is also capable of executing software or firmware implementations of the logic of cache management system 10. However, as will also be apparent to those skilled in the art, the logic of cache management system 10 could also be executed by a web browser at the   contained

12

in the send and receive user station 08 connected to the network, as illustrated by user stations 08 in Figure 1b.

Returning to Figure 1a, cache management system 10 can be used at any or all of the types of sites listed above. For example, if service provider site 00 is used to manage the websites for a number of content providers, service provider site 00 may have its cache management system 10 configured to use a page or data usage frequency cache management method. This could also be related to a charging system that the service provider uses for billing its content providers. Alternatively, cache management system 10 could be configured for a store-through cache management method if the content providers used most frequently rely heavily on the use of interactive forms.

Still in Figure 1a, the administrator of backbone link 04 might prefer to configure its cache management system 10 to use page usage or data usage patterns for providing the best overall response times. As will be apparent to those skilled in the art, all of these administrative decisions and actions could also be done by an expert system dynamically. Similarly, different sites might be configured differently. And also, within one site one set of configurations might apply to one subset of data and a different set to another subset of data.

Local site 06, however, might prefer to use a
time-currency method of cache management. Transmissions over
the Internet using the transmission control protocol/internet
protocol (TP/IP) protocol have date stamps indicating the time
5   at which they were sent, as do many other types of network
protocols. If the information being transmitted is stock quote
data, it is subject to frequent changes during the hours a
given stock exchange is open, but after the close of a trading
day, the closing prices will be valid until the next day of
10   trading on that exchange. If such web-pages are cached using a
"least recently used" method, important stock price changes may
not be brought to a user's attention until that particular
web-page is flushed or replaced in the cache and requires
refreshing from the source. A time-currency method of cache
15   management can be configured to refresh certain pages with one
frequency, say every 15 minutes, during trading hours for a
given exchange, and with another frequency, say until start of
trading the next trading day, once the exchange has closed.

A local site 06 as shown in Figure 1a, might also prefer
20   to use a data usage pattern or even a pre-fetch method of cache
management. This is particularly so where local site 06 is a
corporate firewall/gateway site for an internal network. For
example, if most of the internal users are likely to request

14

pages from the same website, when they first log on, cache management system 10 at local site 06 could be configured to pre-fetch web pages from the requested site each time an internal user logs on and those pages are not already in cache storage. Or, data usage patterns could be tracked and used to manage cache management system 10 on that basis. To illustrate this, if users of a financial journal web page habitually go to a stock quote site when they finish the financial journal pages, this pattern can be combined with pre-fetching of the stock quote pages every time the financial journal pages are fetched. This, in turn, might be coupled with security provisions if access to such pages are to be limited to authorized users only.

When local site 06 is a firewall/gateway site to an internal corporate network, having a number of user sites 08 for its employees, these forms of usage based cache management may be more effective. There may be a greater commonality of interests, and hence data usage among the employees of a corporation, than there would be amongst a disparate grouping of unrelated users.

When local site 06 is a local service provider of dial-up Internet connections for a number of disparate user sites 08,

different types of data usage patterns might be used to manage the cache and charge for services.

Still in Figure 1, content provider site 02 might have still another subset of cache management methods that would work best for it.

Turning now to Figure 2a, an overall flow diagram of the present invention is shown. As shown at step 22, an initial entry is made to the configurator of the present invention. At step 24, the configurator establishes the parameters and other indicators which may be needed by the cache management method(s) selected by the site. As will be apparent to those skilled in the art, a number of methods can be used to indicate which of several options has been selected. In one preferred embodiment, a user supplying the appropriate password might interact with cache management system 10 at each startup or reboot of the site or of a web browser at the site. The options selected by the user may then be indicated by settings or switches in cache management system 10. For simpler cache management algorithms, this may be all that is required.

However, for more complex algorithms, scripts can be prepared for the configurator, supplying additional details of user criteria. Examples of these latter algorithms, with illustrative pseudo-code are shown in Figures 11 and 12.

In an alternative preferred embodiment, the methods to use for cache management can be specified when cache management system 10 is installed at a site.

In yet another embodiment, the methods to be used for cache management at one site could be specified by messages transmitted to it from another site or as a result of messages transmitted to it by a program or script running at the same site, such as a usage pattern analyzer.

For example, such a usage pattern analyzer might track the statistics related to the likelihood that a type of page will already be in the cache when requested. If two methods of cache management are used at the site, pre-fetch for some subsets of data and least recently used (LRU) for others, a pattern analyzer might calculate from history data that the probability of pre-fetch data types being in the cache is .5 versus a lower probability for LRU data. In this instance, preferential treatment would be given to the pre-fetch data when deciding which type should be replaced with new data.

Referring now to Figure 2b, the overall logic of the configurator of the present invention is shown. Here, step 24 from Figure 2a is expanded to show the logic of the configurator. It is essentially a series of decision blocks, for analyzing the data supplied by the operator or by a script

17

or a parameter list or a configuration message. The
configurator, at decision block 24a checks to see if forms will
be handled in a storethrough manner (as described below.) If
yes, processing needed to effectuate that is performed at step
24b and the configurator proceeds next to decision block 24c to
see if data security is to be provided. If yes, processing for
that is done at step 24d. As will be apparent to those skilled
in the art, various types of protection schemes could be
implemented for data that will be stored in the cache, from a
simple scheme, such as password protection, to more elaborate
protections such as encryption.

Returning to the flow in Figure 2b, the system checks, at
decision block 24e, to see whether any kind of indexing cache
management method is selected. If it is, processing for the
indexing method is done at step 24f. Next, the system
determines whether a B-tree structure cache management method
will be used, at decision block 24g. If so, processing for that
is done at step 24h. Still going through Figure 2b, at decision
block 24k, the configurator checks to see whether any pre-fetch
cache management method option has been selected. The
processing at step 241 might include the initial use of a
web-crawler or robot to fetch initial pages. (See description
below for further discussion.)

18

At decision block 24m in Figure 2b, the configurator checks to see whether any timesensitive method of cache management has been selected. If it has, the system may analyze scripted data or parameter data to initialize the values to be used. (See below for use of scripts to supply such data.)

And lastly, in Figure 2b, the configurator checks at step 24o to see if a least recently used cache management method is selected. If it has, then processing associated with it is done at step 24p. If no method has been selected, the configurator can institute a default method, such as LRU. Finally, the configurator logic returns to step 26 in Figure 2a, to proceed with the next tasks.

Now in Figure 2a, once the cache methods selected for the site have been configured, the present invention follows the general flow depicted. At decision block 26, the configurator asks whether data has been requested. If not, the present invention enters a wait state at step 32, until a request comes in. As will be apparent to those skilled in the art, an alternative embodiment could create a task or subtask that is activated only when data requests are made and is suspended at other times.

Again in Figure 2a, if data has been requested, the
configurator checks at decision block 30, to see -if the data
is already in the cache. Depending on the cache management
system used, this step may require either more or less time
than existing systems. If B-tree or indexed caching methods
have been selected, this step may be faster than existing
systems. If time-sensitive methods have been selected, this
step may take longer than existing systems.

If the data requested, usually a web page from a website,
is already in the cache, in this example, storage units 14 the
configurator proceeds to step 27, to supply that data from
storage units 14 in answer to the request, and then to step 28,
to update any indicators associated with the configured cache
management method. Ultimately, it will proceed to step 32, and
wait for the next request.

If, in Figure 2a, at decision block 30 it is determined
that the data is not already in the cache (here, in storage
units 14), a request will be made to fetch the data from the
network at step 34.

At step 36, depending on the cache management method
configured, the indicators, if any, for it will be updated. As
will be described later, if an indexing method is used for
cache management, the index address for storing this data might

be computed at this point, if needed, to reflect a new piece of
data. Then, at step 38, the data is stored in the cache,
storage units 14. It should be noted here, that if the data is
not found because of a failure in storage unit 14, this does
not present a critical reliability problem, since the data can
simply be requested from the network until the failing storage
unit 14 is replaced or repaired.

Turning now to Figure 3, a simple flow diagram of a least
recently used (LRU) method of cache management is shown. When a
new request comes in and the cache is full, as indicated at
step 36a, in Figure 3, the system finds the least recently used
(LRU) data at step 36b and replaces it with the new data at
step 38. Then the system returns to step 32 in Figure 2a, to
wait for the next request.

Figure 4, by contrast, outlines part of the processing for
a time sensitive cache management method. There, once it is
determined at decision block 30 that the data requested is
already in the cache storage unit 14, it is retrieved from
storage unit 14 at step 30a. Then it is checked at step 30b to
see if the time-stamp on the found data is within the
time-stamp parameters configured for this method of cache
management. If it is, then the system provides that data in
answer to the request at step 27c and returns to step 28 in

21

Figure 2a. If the data is not within the time-stamp parameters, a new, fresh copy is requested from the network by going to step 34 in Figure 2a.

An example of time sensitive parameters that can be verified in this way is shown in Figure 12. There scripted parameters CC are specified to indicate that pages are to be kept fresh during the trading hours of a stock exchange. In this example, the opening hours are said to be 1000 hours and the closing hour 1600 hours. During that time, the pages should be refreshed every 15 minutes, according to the scripted amount for value 1. Pseudo-code DD shows how this might be checked at decision block 30b of Figure 4.

A simple variation of the time-sensitive method might include a request that nothing cached be out of date more than some specified period of time. Very little network traffic is generated by simply requesting the version number or creation data of a web page, instead of the entire page or site.

In Figure 5, a flow diagram of a usage-based cache management system 10 is shown. On the Internet, data is found by means of Uniform Resource Locators (URL) addresses. A significant amount of information about usage is thus contained merely in the address of a site. As shown in Figure 13, for example, for domain names, there are several standardized

22

suffixes: com, edu, gov, mil, net, and org. These stand for: commercial, educational, government, military, network service provider, and nonprofit organization, respectively

To illustrate usage based management, a company that markets products to educational institutions might want to give preferential treatment to all educational sites requested by the company's employees. Web pages retrieved from sites having the suffix .edu in their domain names, might be stored with preferential treatment in storage unit 14, so that these pages will not be replaced when the cache is full unless the cache is completely filled with .edu pages. Thus, even though other sites might be more or less frequently used, over time, a cache management system configured in this way will tend to give better response times for requests for .edu pages. As shown in Figure 5, at step 36a, the system configured to use this method of cache management will look for stored data that meets the "not an .edu page" usage requirement to determine where to store a newly retrieved page.

As will be apparent to those skilled in the art, the above use of standard Internet suffixes is illustrative only. Any of a number of other indicators, such as Uniform Resource Locators (URLs) or the identity of the requestor, for a few examples,

23

could also be used in connection with a usage based cache management system 10.

Alternatively, it is also possible that a site might want to track usage first, to establish data patterns by domain name suffix. In Figure 5, this is illustrated at steps 28a, and 36b where usage information is updated. This could be as simple a process as tracking the number of uses of each type of suffix over some specified period. The information gathered from this could be used to change the priorities of caching and replacing data. Other types of usage patterns that might be tracked could relate to images or sound files being referenced by a web page. Figure 14 identifies some of the types of image and sound files that can be included in or referred to in a web page.

In a similar way, information about the request can also be used to pre-fetch data from certain pages or websites. For example, as shown in Figure 13, .information about a particular web-site may be as specific as a "spot" location. A site having a large number of pages may have them individually addressable using the spot address. If a usage study indicates that users of a particular website almost always go from page 1 to pages 14-16, then this information could be configured into the cache management system as shown in Figure 6. If a request meets some pre-fetch criteria, as determined at step 26a in Figure 6,

then an indicator can be set at step 27b to pre-fetch some
specified pages if they are not already in storage unit 14.
These indicators could be automatically checked whenever a
request is made from the network for data not in the cache.

   5   Pre-fetching might also be appropriate for large files
such as image and sound files. As illustrated in Figure 14, a
hypertext reference s1 to a sound file might cause the sound
file to be pre-fetched when the page containing hypertext
reference s1 is retrieved. If frequent accesses are made by all
  10   the users at one site to this web page and all of its hypertext
links, then pre-fetching the files referenced in the hypertext
links will improve response times for such large files as
sound, image and video.

        In much the same way, charging methods of cache management
can be created according to the method and apparatus of the
  15   present invention. An Internet service provider may want to
charge its customers differently for different types of access.
For example, requests for certain classes of domain names could
be charged for differently. Requests for ".com" or commercial
  20   domain names, might be charged a higher rate than requests for
".org" nonprofit sites. If charges are also based on the need
to refresh the cache, the system could track when a request is
made that will cause a request to be made to the network (a

refresh request). This is illustrated in Figure 7, where a
determination is made at step 26a as to whether or not the new
request meets the criteria for changing the charging method.
Thus, if the previous three requests had been for ".com" sites,
and this request is for a ".org" site, and that causes the
system to issue a request to the network, the charge rate would
be changed to that for ".org" and the timed amounts updated.

As will be apparent to those skilled in the art, this
method of cache management could also be combined with the
time-sensitive cache management methods illustrated in Figure
4. Thus, accesses made during the hours a given stock exchange
is open could be billed at a higher rate than those made after
trading hours. In yet another example of a time-sensitive
cache management method, users could be charged for the
"freshness" of the web pages fetched. If the user wants to
insure that all pages of a certain type are less than 7 hours
old, a premium charge could be associated with those requests.

In Figure 8, a flow diagram is shown for using a B-tree
cache management method. B-trees are known to be a fast way to
organize data stored on a disk, so that the disk can be
searched quickly. In a preferred embodiment of the present
invention, if large quantities of storage units 14 are used as
part of cache management system 10, the use of B-trees may be

advantageous for performance purposes. When a new request will result in a store to storage unit 14, the present invention calculates the proper address for the B-tree store at step 36a as shown in Figure 8. In B-trees, a search tree is created of degree n, such that the root node has degree greater than or equal to 2 and every nonterminal node other than the root has degree k, where n/2 is greater than or equal to k and k is greater than or equal to n.

An indexed method of cache management is shown in Figure 9. A very simple index might use the domain names and internet addresses for allocating space and addresses within storage unit 14. As indicated in Figure 9, when a new piece of data comes in, this index can be used to compute, at step 37, the proper address for storing the data in storage unit 14, the cache. When the data is stored in the cache's storage unit 14 at step 38, it is stored at the computed location.

An alternative embodiment of this indexing method might organize the index by the names of frequently accessed image, sound and video files as a top level priority, with other domain names and addresses having a second level priority. In this approach, preference would be given to those files (image, sound or video) that are most likely to require longer transmission times. When data in the cache is to be replaced,

these longer files are not replaced except by other long file types and only after the secondary file types have been replaced.

A number of existing indexing schemes already exist on the Internet for use by programs known as search engines, spiders, web crawlers or robots. When a content provider places a web page on the world wide web, it may also include some index terms in the headers for the website. These indexes are picked up by the search engines and web crawlers when a search request is made over the Internet. An alternative preferred embodiment could use one of these indexing methods to establish the index for the cache management according to the method and apparatus of the present invention. One or more of these web crawlers or robots could also be used in another alternative preferred embodiment to do some or all of the pre-fetching referred to above.

Still another form of indexing or pre-fetching that could be used in an alternative preferred embodiment of the present invention is the technique known as mirroring. If users at a local site are constantly accessing a large website located outside the country, the cache management methods of the present invention might create a local mirror of that site in storage units 14, and use the protocols provided by the source

28

for updating the mirror image. These normally include an initial transfer of all data using a file transfer protocol (FTP) - like protocol, and then regularly scheduled updates that cause any changes made at the source site to be

5   transferred to the mirror. Where the local site has a large amount of storage available for storage units 14, the present invention could include several mirrors in the cache as well as other indexes. Additionally, service providers could offer supplying the mirror files as one of their services. In this

10   approach, updates would be sent to a local site by the service provider as they occur and without being solicited by a file transfer request from the local cache management system 10.

In Figures 1a and 10b, a store-through method of cache management is shown for use with interactive forms such as form f1 shown in Figures 14 and 15. Using any of a number of

15   existing HTML interpreters or parsers (programs that analyze the HTML text present on a page to determine its contents), the configurator checks a d request for the presence of forms at step 26, as shown in Figure 10a. If the data is a form, no

20   check is made to see if it is already in the cache, since it is presumed that forms must be filled out freshly each time. Thus, at step 26a, the check is made to ; if the data contains a form. If it does, the invention goes to step 34 (of Figure 2a)

29

to request that a new copy be transmitted. If the request does
not contain a form, the configurator proceeds to decision block
30 (in Figure 2a) to see if the data is already in the cache.

In the example shown in Figure 15, where the form is a
userid and password verification form, each user at a local
site would fill in a different userid and password, hence
storing one user's filled out form in the cache would be
counterproductive for the other users. Other information that
does not contain forms will be stored through, that is, placed
in the cache according to any other method(s) configured.

In another preferred embodiment, security "doorways" are
provided in cache management system 10, as shown in Figure
10b. Since such security is likely to include the use of some
interactive form, the processing shown in Figure 10a is further
modified to perform the logic shown in Figure 10b. Here, once
it is established that a form is being transmitted, at step
26a, the configurator next checks at step 26e to see if the
form's contents "open" the doorway. First, a check is made at
step 26e to see if the doorway is closed. If it is, at step
26e-1 the entries from the form are checked to see if they are
valid for opening the doorway. If they do -- that is, the
userid and password have been accepted as valid, in this
example -- then that page and those below in the index

30

hierarchy are so marked at step 26f to enable this userid to store and access data in the cache. Once the "doorway" has been opened, the system proceeds to step 26g to exit to step 30 (in Figure 2a) to see if the protected data is already in the cache.

The above described security provisions will work with existing Internet protocols such as http. As will be apparent to those skilled in the art, if the protocols change, or a different protocol is used, the security provisions may need to change as well. In anticipation of such changes, a preferred embodiment would perform the security checking in the cache management system 10, rather than in the applications software used at the site, to minimize the need for other changes.

As will be apparent to those skilled in the art, this or similar forms of security and protection, including such steps as encryption/ decryption for certain pages stored in the cache, may be required by service and content providers who offer to sell goods and services over the internet.

In a preferred embodiment, the logic of the present invention may be embodied in program code written in the C language, either as a software program stored in storage units 14 and executing in control device 12 of cache management system 10, 5 or as firmware executing as part of control device

31

12 of cache management system 10. As will be apparent to those skilled in the art, other programming languages, such as PERL, or Pascal-or C++, or assembler, to name only a few, could be used instead. As mentioned earlier, while it is preferred that

5　　the code execute as part of control device 12 of cache management system 10, it could also be developed to execute as part of a web browser or server manager located at a local site.

Simplified embodiments of the present invention could also be implemented as Unix or Unix shell or Apple Macintosh

10　Applescript scripts that execute in a server operating as one of the links in the network.

As will also be apparent to those skilled in the art, the present invention could also be implemented in hardware

15　circuits using application specific integrated circuits (ASICS) or gate arrays.

While the examples given here are drawn primarily from the Internet network, it will be apparent to those skilled in the art that the apparatus and method of the present invention can

20　be applied to other networks, and similar applications, as well.

Those skilled in the art will appreciate that the embodiments described above are illustrative only and that other systems in the spirit of the teachings herein fall within the scope of the invention.

5   What is claimed as new is: